NAG Library Function Document

nag_outlier_peirce (g07gac)

1 Purpose

nag_outlier_peirce (g07gac) identifies outlying values using Peirce's criterion.

2 Specification

```
#include <nag.h>
#include <nagg07.h>
void nag_outlier_peirce (Integer n, Integer p, const double y[], double mean,
     double var, Integer iout[], Integer *niout, Integer ldiff,
     double diff[], double llamb[], NagError *fail)
```

3 Description

nag_outlier_peirce (g07gac) flags outlying values in data using Peirce's criterion. Let

y denote a vector of n observations (for example the residuals) obtained from a model with p parameters,

m denote the number of potential outlying values,

μ and σ^2 denote the mean and variance of y respectively,

$\tilde{y}$ denote a vector of length $n-m$ constructed by dropping the m values from y with the largest value of $|y_i - \mu|$,

$\tilde{\sigma}^2$ denote the (unknown) variance of $\tilde{y}$,

λ denote the ratio of $\tilde{\sigma}$ and σ with $\lambda = \frac{\tilde{\sigma}}{\sigma}$.

Peirce's method flags y_i as a potential outlier if $|y_i - \mu| \geq x$, where $x = \sigma^2 z$ and z is obtained from the solution of

$$R^m = \lambda^{m-n} \frac{m^m (n-m)^{n-m}}{n^n} \tag{1}$$

where

$$R = 2\exp\left(\left(\frac{z^2-1}{2}\right)(1-\Phi(z))\right) \tag{2}$$

and Φ is the cumulative distribution function for the standard Normal distribution.

As $\tilde{\sigma}^2$ is unknown an assumption is made that the relationship between $\tilde{\sigma}^2$ and σ^2, hence λ, depends only on the sum of squares of the rejected observations and the ratio estimated as

$$\lambda^2 = \frac{n-p-mz^2}{n-p-m}$$

which gives

$$z^2 = 1 + \frac{n-p-m}{m}\left(1-\lambda^2\right) \tag{3}$$

A value for the cutoff x is calculated iteratively. An initial value of $R = 0.2$ is used and a value of λ is estimated using equation (1). Equation (3) is then used to obtain an estimate of z and then equation (2) is used to get a new estimate for R. This process is then repeated until the relative change in z between consecutive iterations is $\leq \sqrt{\epsilon}$, where ϵ is ***machine precision***.

By construction, the cutoff for testing for $m+1$ potential outliers is less than the cutoff for testing for m potential outliers. Therefore Peirce's criterion is used in sequence with the existence of a single potential outlier being investigated first. If one is found, the existence of two potential outliers is investigated etc.

If one of a duplicate series of observations is flagged as an outlier, then all of them are flagged as outliers.

4 References

Gould B A (1855) On Peirce's criterion for the rejection of doubtful observations, with tables for facilitating its application *The Astronomical Journal* **45**

Peirce B (1852) Criterion for the rejection of doubtful observations *The Astronomical Journal* **45**

5 Arguments

1: **n** – Integer *Input*

On entry: n, the number of observations.

Constraint: $\mathbf{n} \geq 3$.

2: **p** – Integer *Input*

On entry: p, the number of parameters in the model used in obtaining the y. If y is an observed set of values, as opposed to the residuals from fitting a model with p parameters, then p should be set to 1, i.e., as if a model just containing the mean had been used.

Constraint: $1 \leq \mathbf{p} \leq \mathbf{n}-2$.

3: **y**[**n**] – const double *Input*

On entry: y, the data being tested.

4: **mean** – double *Input*

On entry: if **var** > 0.0, **mean** must contain μ, the mean of y, otherwise **mean** is not referenced and the mean is calculated from the data supplied in **y**.

5: **var** – double *Input*

On entry: if **var** > 0.0, **var** must contain σ^2, the variance of y, otherwise the variance is calculated from the data supplied in **y**.

6: **iout**[**n**] – Integer *Output*

On exit: the indices of the values in **y** sorted in descending order of the absolute difference from the mean, therefore $|\mathbf{y}[\mathbf{iout}[i-2]-1]-\mu| \geq |\mathbf{y}[\mathbf{iout}[i-1]-1]-\mu|$, for $i = 2,3,\ldots,\mathbf{n}$.

7: **niout** – Integer * *Output*

On exit: the number of potential outliers. The indices for these potential outliers are held in the first **niout** elements of **iout**. By construction there can be at most $\mathbf{n}-\mathbf{p}-1$ values flagged as outliers.

8: **ldiff** – Integer *Input*

On entry: the maximum number of values to be returned in arrays **diff** and **llamb**.

If **ldiff** ≤ 0, arrays **diff** and **llamb** are not referenced and both **diff** and **llamb** may be **NULL**.

9: **diff**[**ldiff**] – double *Output*

On exit: if **diff** is not **NULL** then **diff**$[i-1]$ holds $|y-\mu|-\sigma^2 z$ for observation **y**[**iout**$[i-1]-1$], for $i = 1, 2, \ldots, \min(\mathbf{ldiff}, \mathbf{niout}+1, \mathbf{n}-\mathbf{p}-1)$.

10: **llamb**[**ldiff**] – double *Output*

On exit: if **llamb** is not **NULL** then **llamb**$[i-1]$ holds $\log\left(\lambda^2\right)$ for observation **y**[**iout**$[i-1]-1$], for $i = 1, 2, \ldots, \min(\mathbf{ldiff}, \mathbf{niout}+1, \mathbf{n}-\mathbf{p}-1)$.

11: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 2.3.1.2 in How to Use the NAG Library and its Documentation for further information.

NE_BAD_PARAM

On entry, argument ⟨*value*⟩ had an illegal value.

NE_INT

On entry, $\mathbf{n} = \langle value \rangle$.
Constraint: $\mathbf{n} \geq 3$.

NE_INT_2

On entry, $\mathbf{p} = \langle value \rangle$ and $\mathbf{n} = \langle value \rangle$.
Constraint: $1 \leq \mathbf{p} \leq \mathbf{n} - 2$.

NE_INTERNAL_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please contact NAG for assistance.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 2.7.6 in How to Use the NAG Library and its Documentation for further information.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 2.7.5 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

Not applicable.

8 Parallelism and Performance

nag_outlier_peirce (g07gac) is not threaded in any implementation.

9 Further Comments

One problem with Peirce's algorithm as implemented in nag_outlier_peirce (g07gac) is the assumed relationship between σ^2, the variance using the full dataset, and $\tilde{\sigma}^2$, the variance with the potential outliers removed. In some cases, for example if the data y were the residuals from a linear regression, this assumption may not hold as the regression line may change significantly when outlying values have been dropped resulting in a radically different set of residuals. In such cases nag_outlier_peirce_two_var (g07gbc) should be used instead.

10 Example

This example reads in a series of data and flags any potential outliers.

The dataset used is from Peirce's original paper and consists of fifteen observations on the vertical semidiameter of Venus.

10.1 Program Text

```
/* nag_outlier_peirce (g07gac) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

/* Pre-processor includes */
#include <stdio.h>
#include <math.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg07.h>

int main(void)
{

  /* Integer scalar and array declarations */
  Integer i, n, niout, p, exit_status, ldiff;
  Integer *iout = 0;

  /* NAG structures */
  NagError fail;

  /* Double scalar and array declarations */
  double *y = 0, *diff = 0, *llamb = 0;

  /* Let the routine calculate the mean and variance from the supplied data */
  double mean = 0.0;
  double var = 0.0;

  exit_status = 0;

  /* Initialize the error structure */
  INIT_FAIL(fail);

  printf("nag_outlier_peirce (g07gac) Example Program Results\n");

  /* Skip headings in data file */
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif

  /* Read in the problem size */
#ifdef _WIN32
```

```
  scanf_s("%" NAG_IFMT " %" NAG_IFMT " %" NAG_IFMT "%*[^\n] ", &n, &p,
          &ldiff);
#else
  scanf("%" NAG_IFMT " %" NAG_IFMT " %" NAG_IFMT "%*[^\n] ", &n, &p, &ldiff);
#endif

  if (!(y = NAG_ALLOC(n, double)) ||
      !(diff = NAG_ALLOC(ldiff, double)) ||
      !(llamb = NAG_ALLOC(ldiff, double)) || !(iout = NAG_ALLOC(n, Integer)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }

  /* Read in the data */
  for (i = 0; i < n; i++)
#ifdef _WIN32
    scanf_s("%lf%*[^\n] ", &y[i]);
#else
    scanf("%lf%*[^\n] ", &y[i]);
#endif

  /* Use nag_outlier_peirce (g07gac) to get a list of potential outliers */
  nag_outlier_peirce(n, p, y, mean, var, iout, &niout, ldiff, diff, llamb,
                     &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_outlier_peirce (g07gac).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
  }

  /* Display the potential outliers */
  printf("\n");
  printf(" Number of potential outliers: %" NAG_IFMT "\n", niout);
  if (ldiff > 0) {
    printf(" %5s  %5s %8s   %8s    %8s\n", "No.", "Index", "Value",
           "Diff", "ln(lambda^2)");
  }
  else {
    printf(" %5s  %5s %8s\n", "No.", "Index", "Value");
  }
  for (i = 0; i < niout; i++)
    if (i < ldiff) {
      printf("%5" NAG_IFMT " %5" NAG_IFMT " %10.2f %10.2f %10.2f\n", i + 1,
             iout[i], y[iout[i] - 1], diff[i], llamb[i]);
    }
    else {
      printf("%5" NAG_IFMT " %5" NAG_IFMT " %10.2f\n", i + 1, iout[i],
             y[iout[i] - 1]);
    }

END:
  NAG_FREE(y);
  NAG_FREE(diff);
  NAG_FREE(llamb);
  NAG_FREE(iout);

  return exit_status;
}
```

10.2 Program Data

```
nag_outlier_peirce (g07gac) Example Program Data
15 2 1 :: n,p,ldiff
-0.30
 0.48
 0.63
-0.22
 0.18
```

```
-0.44
-0.24
-0.13
-0.05
 0.39
 1.01
 0.06
-1.40
 0.20
 0.10  :: y
```

10.3 Program Results

```
nag_outlier_peirce (g07gac) Example Program Results

 Number of potential outliers: 2
   No.  Index    Value       Diff    ln(lambda^2)
    1    13      -1.40       0.31      -0.30
    2    11       1.01
```